03-06-06

AFW

PTO/SB/21 (08-00)
Approved for use through 10/31/02. OMB 0651-0031
Patent and Trademark Office: U.S. DEPARTMENT OF COMMENRCE
Under the Paper Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays avalid OMB control number.

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Application Number	10/040,578
Filing Date	December 28, 2001
First Named Inventor	David J. Long
Group Art Unit	2192
Examiner Name	PHAM, CHRYSTINE
Total Number of Pages in This Submission: 28	Attorney Docket Number: 50277-1766

ENCLOSURES ***(check all that apply)***

- [X] Fee Transmittal Form
  - [X] Fee Attached
- [ ] Amendment / Response
  - [ ] After Final
  - [ ] Affidavits/declaration(s)
- [ ] Extension of Time Request
- [ ] Express Abandonment Request
- [ ] Information Disclosure Statement
- [ ] Certified Copy of Priority Document(s)
- [ ] Response to Missing Parts/ Incomplete Application
  - [ ] Response to Missing Parts under 37 CFR 1.52 or 1.53
- [ ] Assignment Papers *(for an Application)*
- [ ] Drawing(s)
- [ ] Licensing-related Papers
- [ ] Petition
- [ ] Petition To Convert To a Provisional Application
- [ ] Power of Attorney, Revocation Change of Correspondence Address
- [ ] Terminal Disclaimer
- [ ] Request for Refund
- [ ] CD, number of CD(s)_____
- [ ] After Allowance Communication to Group
- [ ] Appeal Communication to Board of Appeals and Interferences
- [X] Appeal Communication to Group *(Appeal Notice, Brief, Reply Brief)*
- [ ] Proprietary Information
- [ ] Status Letter
- [ ] Other Enclosure(s) *(please identify below):*

Remarks

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm *or* Individual name	Hickman Palermo Truong & Becker LLP Stephen J. Shaw, Reg. No. 56,442
Signature	
Date	March 3, 2006

CERTIFICATE OF MAILING

Express Mail" mailing label number EV 861821695 US Date of Deposit: March 3, 2006

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to Mail Stop Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Type or printed name	Jennifer Newell		
Signature		Date	March 3, 2006

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time your are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450

PTO/SB/17 (12/04)
Approved for use through 09/30/2005. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

FEE TRANSMITTAL for FY 2006

Patent fees are subject to annual revision, Small Entity payments must be supported by a small entity statement, otherwise large entity fees must be paid. See Forms PTO/SB/09-12. See 37 C.F.R. §§ 1.27 AND 1.28

TOTAL AMOUNT OF PAYMENT	($) 620.00

Complete if Known	
Application Number	10/040,578
Filing Date	December 28, 2001
First Named Inventor	David J. Long
Examiner Name	Pham, Chrystine
Group/Art Unit	2192
Attorney Docket No.	50277-1766

METHOD OF PAYMENT (check one)

1. ☒ Throughout the pendency of this application, please charge any additional fees, including any required extension of time fees, and credit all overpayments to deposit account 50-1302. A duplicate of this sheet is enclosed.

Deposit Account Number: 50-1302

Deposit Account Name: Hickman Palermo Truong & Becker, LLP

2. ☒ Payment Enclosed:

☒ Check ☐ Money Order ☐ Other

3. ☐ Applicant(s) is entitled to small entity status. See 37 CFR 1.27.

FEE CALCULATION

1. BASIC FILING FEE

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
1011	300	2011	150	Utility filing fee	
1111	500	2111	250	Utility Search fee	
1311	200	2311	100	Utility Examination fee	
1081	250	2081	125	Utility Application Size Fee	
1005	200	2005	100	Provisional Application Fee	
1085	250	20835	125	Provisional Application Size Fee	
				SUBTOTAL (1)	($) 0.00

2. EXTRA CLAIM FEES

	Highest Paid Claims		Extra Claims		Fee from Below		Fee Paid
Total Claims		-20**=	0	x	50.00	=	0.00
Independent Claims		- 3**=	0	x	200.00	=	0.00
Multiple Dependent						=	

**or number previously paid, if greater; For Reissues, see below

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description
1202	50	2202	25	Claims in excess of 20
1201	200	2201	100	Independent claims in excess of 3
1203	360	2203	180	Multiple dependent claim, if not paid
1204	200	2204	100	**Reissue independent claims over original patent
1205	50	2205	25	**Reissue claims in excess of 20 and over original patent
				SUBTOTAL (2) ($) 0.00

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
1051	130	2051	65	Surcharge – late filing fee or oath	
1052	50	2052	25	Surcharge – late provisional filing fee or cover sheet.	
1251	120	2251	60	Extension for reply within first month	120.00
1252	450	2252	225	Extension for reply within second month	
1253	1,020	2253	510	Extension for reply within third month	
1254	1,590	2254	795	Extension for reply within fourth month	
1255	2,160	2255	1,080	Extension for reply within fifth month	
1401	500	2401	250	Notice of Appeal	
1402	500	2402	250	Filing a brief in support of an appeal	500.00
1452	500	2452	250	Petition to revive – unavoidable	
1453	1,500	2453	750	Petition to revive – unintentional	
1501	1,400	2501	700	Utility issue fee (or reissue)	
1502	800	2502	400	Design issue fee	
1504	300	2504	300	Publication Fee	
1462	400	1462	400	Petitions Director not specifically provided for Group I	
1463	200	1463	200	Petitions Director not specifically provided for Group II	
1464	130	1464	130	Petitions Director not specifically provided for Group III	
1806	180	1806	180	Submission of information Disclosure Stmt	
8021	40	8021	40	Recording each patent assignment per property (times number of properties)	
1809	790	2809	395	Filing a submission after final rejection (37 CFR § 1.129(a))	
1810	790	2810	395	For each additional invention to be examined (37 CFR § 1.129(b))	
				Other fee (specify)	
				Other fee (specify)	

*Reduced by Basic Filing Fee Paid SUBTOTAL (3) ($) 620.00

SUBMITTED BY

Name (Print/Type)	Stephen J. Shaw	Registration No. (Attorney/Agent)	56,442	Telephone	(408) 414-1080
Signature				Date	March 3, 2006

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231.
DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Mail Stop Amend, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re application of:

David J. Long, et al.

Serial No.: 10/040,578

Filed on: December 28, 2001

For: PROPERTY BUNDLES ON A PER-INSTANCE BASIS

Confirmation No.: 3641

Examiner: Pham, C.

Group Art Unit No.: 2192

MS Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

This Appeal Brief is submitted in support of the Notice of Appeal filed on November 14, 2005. Appellants filed a Request for Pre-Appeal Brief Review with the Notice of Appeal. A Notice of Panel Decision from Pre-Appeal Brief Review was mailed on January 3, 2006. Therefore, the time period for filing this Appeal Brief extends to February 3, 2006. A petition and fee for a one (1) month extension of time is hereby enclosed.

I. REAL PARTY IN INTEREST

Oracle International Corporation is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

Appellants are unaware of any related appeals or interferences.

III. STATUS OF CLAIMS

Claims 1-32 have been finally rejected and are the subjects of this appeal.

IV. STATUS OF AMENDMENTS

Claims 1-32 were not amended after the Final Office Action.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The present application contains independent Claims 1 and 17, which are summarized below. Claims 2-6, 10-13, 17-22 and 26-29 are argued separately from the independent claims upon which they depend, and therefore are summarized below. The remaining dependent claims are not argued separately from the independent claims upon which they depend, and therefore are not summarized below.

The claims summarized below are annotated to cross-reference features of the claims to specific examples of those features disclosed in the specification. However, the annotations are not intended to limit the scope of the recited features to those specific examples to which the annotations refer.

Claim 1 recites (with added reference annotations in parenthesis) a method of specifying a structure, within a computer system, (FIG. 11, computer system 1100), of an instance (FIG. 7a, Document Object 700) of a class (FIG. 2a, Document Class 220), the method including the step of:

associating with said instance (FIG. 7a, Document Object 700) of said class (FIG. 2a, Document Class 220) an attribute (FIG. 7a, Number of Tracks) that is not in said class (FIG. 2a, Document Class 220) or any superclass of said class (FIG. 2a, Public_Object Class), thereby

establishing for said instance (FIG. 7a, Document Object 700) said structure that includes storage for data associated with said attribute (FIG. 7b, 27); and

wherein associating said attribute (FIG. 7a, Number of Tracks) with said instance (FIG. 7a, Document Object 700) does not cause said attribute to become an attribute of said class (FIG. 2a, Document Class 220).

Claim 2 recites (with added reference annotations in parenthesis) a method similar to that of Claim 1, wherein the step of associating further includes the steps of:

establishing a property bundle that is associated with one or more attributes that are not in said class or any super class of said class (FIG. 3a, Property Bundle 301); and

associating said instance with said property bundle (pages 16, line 19, to page 17, line 22).

Claim 3 recites (with added reference annotations in parenthesis) a method similar to that of Claim 2, further comprising the step of:

storing within a database, objects that define said instance, said property bundle, and said one or more attributes (page 14, lines 1-21).

Claim 4 recites (with added reference annotations in parenthesis) a method similar to that of Claim 1, further comprising the step of:

maintaining an object relational mapping system that indicates a correlation between said instance and data stored in a relational database (page 14, lines 1-21, page 17, lines 1-8).

Claim 5 recites (with added reference annotations in parenthesis) a method similar to that of Claim 1, wherein the step of associating includes establishing a pointer from said instance to a property bundle (page 16, lines 19-23).

Claim 6 recites (with added reference annotations in parenthesis) a method similar to that of Claim 5, wherein the step of associating includes establishing a pointer from said attribute to said property bundle (page 17, lines 1-8).

Claim 10 recites (with added reference annotations in parenthesis) a method similar to that of Claim 1, wherein the step of associating further includes the step of:

maintaining a table that includes an entry that indicates that said instance is associated with said attribute (page 25, lines 3-20).

Claim 11 recites (with added reference annotations in parenthesis) a method similar to that of Claim 10, wherein the step of maintaining a table further includes the step of:

maintaining said entry to include a key that identifies said attribute (page 25, lines 3-20).

Claim 12 recites (with added reference annotations in parenthesis) a method similar to that of Claim 10, wherein the step of maintaining a table further includes the step of:

maintaining said table externally to said instance (page 14, lines 1-21, page 25, lines 3-20).

Claim 13 recites (with added reference annotations in parenthesis) a method similar to that of Claim 10, wherein the step of maintaining a table further includes the step of:

maintaining said table internally to said instance (page 14, lines 1-21, page 25, lines 3-20).

Claims 17-31 recite computer readable media (FIG. 11, storage device 1110) that carry instructions for causing processors (FIG. 11, processor 1104) to perform (page 30, lines 16-22) the steps of the methods of Claims 1-16, respectively.

The methods, systems, and media of the above claims allow methods and attributes to be associated with instances of classes on a per-instance basis. The associating of the properties with objects on a per-instance basis may be used in any of the following ways:

1) different instances of the same class are associated with different properties where the properties are not in the class; and

2) two instances of two different classes are associated with the same property where the property is not in either of the two classes.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Claims 1-15 and 17-31 stand rejected under 35 U.S.C. § 102(e) as being anticipated, allegedly, by U.S. Patent No. 6,385,618 B1 to Ng et al. ("*Ng*").

2. Claims 16 and 2 stand rejected under 35 U.S.C. § 103(a) as being unpatentable, allegedly, over *Ng* in view of The Java Virtual Machine Specification (The class File Format, http://java.sun.com/docs/books/vmspec/html/ClassFile.doc.html) ("*JVM Spec*").

VIII. ARGUMENTS

A. The Features of Claims 1 and 17 Aren't Taught or Suggested by Ng

1. *Ng does not disclose, teach or suggest associating with an instance of a class an attribute that is not in the class or any superclass of the class*

One of the reasons that the embodiments of the invention recited in Claims 1-32 are advantageous is because they allow methods and attributes to be associated with instances of classes on a per-instance basis. Previously, in Object Oriented programming it was not possible for an object to have a different set of attributes from another object where the objects were instances of the same class. Nor was it possible for an object to have a different implementation of a method than another object, where the objects were instances of the same class. This advantage is captured, for example, in the following language of Claim 1:

"associating with said instance of said class **an attribute that is not in said class or any superclass of said class**...."

Ng does not teach or suggest these embodiments of the invention. In support of the §102(e) rejection based upon *Ng* allegedly anticipating the above-referenced feature, the Examiner relies solely upon the assertion that the data member "*collection Orders_for_Customer*" of "Class Customer 420" in Fig. 4B and "*int Cust_*id" are "attribute[s] that are **not in** said class [the "Customer" class] or any superclass of said class."

The "Customer" class described in *Ng* and relied upon by the Examiner is illustrated thusly in Figure 4B:

```
Class Customer {

        int Cust_id;
        str SSN;
        collection      Orders_for_Customer;
        int             get_Cust_id();
        void            set_Cust_id(int Cust_id);
        str             get_SSN();
        void            set_SSN(str SSN);
        iterator        getOrdersForCustomer();
        void            addOrdersForCustomer(Order O);
        void            removeOrdersForCustomer(Order O);
}
```

(*Ng*, Fig. 4B) (emphasis added)

As is now obvious, the data members (*i.e.*, attributes) "*collection Orders_for_Customer*" and "*int Cust_*id" clearly are **attributes that ARE in the class** (Customer), just as "*str SSN*" is an attribute of the Customer class.

Ng makes this clear in the text associated with FIG. 4B:

> Class 420 reflects customer table 202 and class 424 reflects order table 204. As such, class 420 ***contains*** **a data member for customer ID, social**

> **security number, and a collection of objects representing the orders associated with that particular customer**, thus implementing the foreign key. Class 420 also contains a number of methods to both get and set the value of the data members, including an iterator method to iterate through the order for this particular customer.
>
> ...
>
> To define this relationship in the Java ™ programming language, the class representing the referring table is defined **to *have a member* that is a collection of the class representing the referred table**. A "collection" refers to a type indicating a grouping of instances of other classes. Then, ***in the class* reflecting the referred table, a *member is added* providing a reference to the class that refers to it**.

(*Ng*, col. 6, lines 37-66)

Because the data members (*i.e.*, attributes) "*collection Orders_for_Customer*" and "*int Cust_id*" clearly **ARE** in the Customer class, the reference in no way anticipates the feature at issue, which requires "associating with said instance of said class an **attribute** that **is not in said class** or any superclass of said class." As is obvious from the reference, an instance (object) instantiated from the Customer class in *Ng* will contain the attributes "*collection Orders_for_Customer*" and "*int Cust_id*" and will NOT contain any attributes contained in another class. Conversely, no instance of the Customer class in *Ng* will contain an attribute NOT in the Customer class. There is simply no teaching or suggestion in *Ng* to demonstrate otherwise.

This becomes even clearer when FIG. 4B is analyzed in the context of *Ng*. *Ng* is directed to an "object-relational mapping tool... that generates source code containing class which preserve both changes to the database schema as well as customizations to a preexisting version of the classes." (*Ng*, Col. 3, lines 33-38) The technique disclosed in *Ng* "maps a database by first querying the database to determine its schema and then by creating an internal data structure... representing that schema. From this data structure, the object-relational mapping tool creates an object model containing all the information necessary to generate

classes and then creates source code containing a number of Java classes that may be used by a programmer to interface with a database." (*Ng, c*ol. 4, lines 23-31)

Nowhere in *Ng* may be found structures or techniques capable of associating with an instance of a class **an attribute that is not in said class or any superclass of said class**. Just the opposite: *Ng* generates standard Java classes consistent with a database schema. As previously discussed, prior to the techniques and embodiments of the present Application, it was not possible in standard object-oriented programming (Java) to perform the feature in Claim 1, namely, "associating with said instance of said class an attribute that **is not in said class** or any superclass of said class." Certainly, nothing in *Ng* allows for the possibility.

"A claim is anticipated under § 102 only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." *Verdegaal Bros. v. Union Oil Co. of California,* 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987).

There is no "associating with said instance of said class an attribute that **is not in said class** or any superclass of said class" in the reference that could conceivably anticipate or in any way suggest the act of "associating with said instance of said class an attribute that **is not in said class** or any superclass of said class."

The Examiner relies upon a single figure and its associated text to support the § 102(e) rejection. As is now obvious, the figure and the associated text not only directly contradicts the Examiner's assertions, but also demonstrates that it is impossible for *Ng* to perform, teach or suggest the claimed feature and remain consistent with its own teachings.

As a result, Claims 1 and 17 are patentable over *Ng* under 35 U.S.C. § 102(e). The rejection of Claims 1 and 17 should be reversed.

2. *Ng does not disclose, teach or suggest associating an attribute with an instance where the association does not cause the attribute to become an attribute of the class*

Claim 1 reads, in pertinent part:

> associating with said instance of said class an attribute that is not in said class or any superclass of said class, thereby establishing for said instance said structure that includes storage for data associated with said attribute; and
>
> **wherein associating said attribute with said instance does not cause said attribute to become an attribute of said class.**

Therefore, the highlighted feature requires associating an "attribute that is not in said class or any superclass of said class" with an instance of a class, where the association does not cause the attribute to become an attribute of the class from which the instance is created. The Examiner cites element Class 420 of FIG. 4, element customer table 202 of FIG. 2 and col. 6, lines 30-67 of *Ng* as allegedly disclosing this feature.

As discussed above in section (1), the Examiner cited virtually the same text and Figure as allegedly anticipating the step of, "associating with said instance of said class an attribute that **is not in said class** or any superclass of said class." As has previously been shown, this reliance is misplaced, as the cited text actually contradicts the Examiner's assertion rather than supports it.

Therefore, since no portion of *Ng* can anticipate the claimed "associating with said instance of said class an attribute that **is not in said class** or any superclass of said class," no portion of *Ng* can anticipate the claimed, "wherein **associating said attribute** with said instance does not cause said attribute to become an attribute of said class" because this feature requires associating an attribute not in a class with an instance of that class. As a result, Claims

1 and 17 are patentable over *Ng* under 35 U.S.C. § 102(e). The rejection of Claims 1 and 17 should be reversed.

Even assuming *arguendo* that *Ng* anticipates the claimed "associating with said instance of said class an attribute that **is not in said class** or any superclass of said class," *Ng* does not anticipate the step of "wherein **associating said attribute** with said instance does not cause said attribute to become an attribute of said class" and the rejection should still be reversed.

This conclusion becomes apparent upon close inspection of the portions of *Ng* relied upon by the Examiner. The cited figure elements and section of *Ng* reads: (with the specific portions cited by the Examiner highlighted):

> [s]uch as hash tables, one skilled in the art will appreciate that other data structures may also be used.
>
> As can be appreciated from this description of object model 116, it contains all of the information necessary to create the classes in the source code, an example of which is depicted in FIG. 4B. FIG. 4B depicts source code file 116 with the Java.TM. programming language representation of objects 400 and 401. **Class 420** reflects **customer table 202** and class 424 reflects order table 204. As such, **class 420** contains a data member for customer ID, social security number, and a **collection of objects** representing the **orders** associated with that **particular customer**, thus implementing the foreign key. **Class 420** also contains a number of methods to both get and set the value of the data members, including an iterator method to iterate through the order for this particular customer. Class 424 includes data members order ID and date and also includes various methods to both set and get the values for these data members. Additionally, class 424 contains a field, Customer_for_Order, implementing the foreign key with a reference to the particular customer object that placed that order.
>
> When a foreign key is contained in the object model, the object-relational mapping tool typically creates a relationship in the source code between two classes to implement the foreign key. As stated above, with a foreign key, one or more records in one table (the referring table) refers to one record in another table (the referred table). This relationship is a one-to-many relationship, although it may be a one-to-one relationship. Additionally, instead of being bidirectional, the relationship may be unidirectional. To define this relationship in the Java.TM. programming language, the class representing the referring table is defined to have a member that is a collection of the class

> representing the referred table. A "collection" refers to a type indicating a **grouping of instances** of other classes. Then, in the class reflecting the referred table, a member is added providing a reference to the class that refers to it. For most cases, this is how a foreign key is implemented. However, when the foreign key for two....

(*Ng*, col. 6, lines 30-67)(emphasis added)

While the Examiner did not clearly explain the pertinence of the reference, instead citing a large portion of text and two figure elements, Applicant will attempt to extrapolate the Examiner's position from the cited elements. It appears that the Examiner asserts that a collection is an attribute of a class, and because the collection indicates a "grouping of instances of other classes," the "grouping of instances" is equivalent to an attribute of another class.

This is incorrect. When the cited example is viewed in light of the entire specification, the true meaning becomes apparent. As stated earlier, *Ng* is directed to techniques for generating standard Java classes consistent with a database schema. This is reflected in the above cited text by "**Class 420** reflects **customer table 202** and class 424 reflects order table 204." Customer table 202 and order table 204 are tables in a database. "FIG. 2 depicts a more detailed diagram of an example of database 118, containing a customer table 202 and an order table 204." (*Ng*, col. 5, lines 37-39)

The customer table 202 contains a customer ID column 206, which serves as the primary key for the customer table 202. (*Ng*, col. 5, lines 39-42). The order table 204 contains a customer ID column 216, which serves as the foreign key to the customer ID column 206, thereby relating the customer ID column 216 in the order table 204 to the customer ID column 206 in the customer table 202. (*Ng*, col. 5, lines 42-47). This is basic database functionality allowing an order to be linked to a specific customer. The techniques in *Ng* take the database schema and create standard Java classes consistent with the database schema; therefore, the

class 420 reflects customer table 202 and class 424 reflects order table 204. To implement the foreign key, *Ng* creates a collection attribute called Orders_for_Customer. The collection attribute simply references instances of the class 424. These instances reflect orders placed by the specific customer that is represented by the instance of the class 420.

Nowhere is there any reference, teaching or suggestion that the act of associating a collection attribute with an **instance** of the Customer Class 420 does not cause the collection attribute to become an attribute of the Customer Class 420. The collection attribute **IS** an attribute of the Customer Class 420, just like *int Cust_id* and *str SSN*. Instances of the Customer Class 420 will have the attributes *collection Orders_for_Customer*, *int Cust_id* and *str SSN*. It appears that the Examiner is equating what the collection attribute **references** (objects of another class) with the attribute itself. This is akin to stating that a variable (*int Cust_id*) is equivalent to the value of the variable, which is obviously incorrect and in any event does not anticipate the claim element at issue.

Therefore, Claims 1 and 17 are patentable over *Ng* under 35 U.S.C. § 102(e). The rejection of Claims 1 and 17 should be reversed.

B. The Features of Claims 2 and 18 Aren't Taught or Suggested by *Ng*

By virtue of its dependence from Claim 1, Claim 2 includes the features of Claims 1 that are distinguished from *Ng*. Therefore, Claim 2 is patentable over *Ng*.

Additionally, Claim 2 recites the feature, "establishing a property bundle **that is associated with one or more attributes that are not in said class or any super class of said class**; and associating said instance with said property bundle." *Ng* does not teach or suggest this feature.

The Examiner again cites elements of Figure 4B and associated text of *Ng*, as well as elements of Figure 3 and associated text. Again, while the Examiner did not clearly explain the pertinence of the cited portions of the reference, instead citing a large portion of text and various figure elements, Applicant will attempt to extrapolate the Examiner's position from the cited elements. It appears that the Examiner asserts that the attributes *collection Orders_for_Customer* and *int Cust_id* are equivalent to a property bundle that is associated with attributes that are not in the Customer Class 420, such as *customer* 302, *foreign key* 306 and *order* 304.

The Examiner apparently analogizes the elements (*customer* 302, *foreign key* 306 and *order* 304) to attributes. An examination of *Ng* indicates this is incorrect. Element 302 actually refers to an object model reflecting the customer table 202 in a database schema. Element 304 refers to an object model reflecting the order table 204 in a database schema. Element 306 refers to a list of objects referencing foreign keys.

> Once created, database data structure 115 appears as shown in FIG. 3 and includes an object 302, reflecting the customer table 202, and an object 304, reflecting the order table 204. Object 302 contains a list 306 of foreign key objects, if any, each containing the name of the foreign key as well as an indication of the columns that comprise the foreign key.

(*Ng*, col. 5, lines 54-59)

The above section does not appear to say anything about attributes at all.

Therefore, *Ng* fails to disclose multiple features of Claim 2.

Claim 18 recites a computer-readable medium that carries instructions for performing the steps of the method of Claim 2, including the features distinguished from *Ng*.

As a result, Claims 2 and 18 are patentable over *Ng* under 35 U.S.C. § 102(e). The rejection of Claims 2 and 18 should be reversed.

C. The Features of Claims 3 and 19 Aren't Taught or Suggested by *Ng*

By virtue of its dependence from Claim 2, Claim 3 includes the features of Claim 2 that are distinguished from *Ng*. Therefore, Claim 3 is patentable over *Ng*.

Additionally, Claim 3 recites the feature, "**storing within a database, objects that define said instance, said property bundle, and said one or more attributes**."

The Examiner cites *Ng*, col. 1, line 60 to col. 2, line 11 as allegedly disclosing the storing of objects that define said instance, said property bundle, and said one or more attributes. The lines in this cited section of *Ng* read, in total:

> Object-relational mapping tools have been created to facilitate development of application programs that utilize a relational database. A relational database stores data in tables having rows (records) and columns (fields). The tables are usually interrelated, and thus, there is a logical structure imposed on the database. This logical structure is known as a schema. Each table may have a primary key, comprising one or more columns that uniquely identify a row. For example, in a table with rows of customers, a column storing each customer's social security number may be used as the primary key because it uniquely identifies each customer in the table. A table may also have one or more foreign keys, associating a row in one table to one or more rows in another table. For example, where one table contains customer information and another table contains order information for the customers, a foreign key may exist in the order table to relate one customer (or row) in the customer table with one or more orders (or rows) in the order table.

There appears to be nothing in the cited section that would correspond to the "storing within a database, objects that define said instance, said property bundle, and said one or more attributes" of Claim 3. The only time the word "object" is mentioned in the cited section is referring to some amorphous previously-created "Object-relational mapping tools." The rest of the cited section simply discusses characteristics of a relational database. If there is some information discussed in this section that might disclose, teach or suggest the act of storing **objects** in a relational database, it is not apparent. Simply having the word "object" in a

paragraph discussing relational databases is a far cry from actually anticipating the use of a relational database to store objects that define instances, property bundles, and attributes.

Therefore, *Ng* fails to disclose multiple features of Claim 3.

Claim 19 recites a computer-readable medium that carries instructions for performing the steps of the method of Claim 3, including the features distinguished from *Ng*.

As a result, Claims 3 and 19 are patentable over *Ng* under 35 U.S.C. § 102(e). The rejection of Claims 3 and 19 should be reversed.

D. The Features of Claims 4 and 20 Aren't Taught or Suggested by Ng

By virtue of its dependence from Claim 1, Claim 4 includes the features of Claim 1 that are distinguished from *Ng*. Therefore, Claim 4 is patentable over *Ng*.

Additionally, Claim 4 recites the feature, "**maintaining an object relational mapping system that indicates a correlation between said instance and data stored in a relational database.**"

The Examiner again cites *Ng*, col. 1, line 60 to col. 2, line 11 as allegedly disclosing this feature. As discussed above with regard to Claim 3, there appears to be nothing in the cited section that would correspond to maintaining an object relational mapping system that indicates a correlation between an instance (of an object) and data stored in a relational database.

The only time the word "object" is mentioned in the cited section is referring to some amorphous previously-created "Object-relational mapping tools." The rest of the cited section simply discusses characteristics of a relational database. If there is some information discussed in this section that might disclose, teach or suggest the act of indicating a correlation between an instance (of an object) and data stored in a relational database, it is not apparent. Simply

having the word "object" in a paragraph discussing relational databases is a far cry from actually anticipating the use of a relational database to store objects that define instances, property bundles, and attributes.

Therefore, *Ng* fails to disclose multiple features of Claim 4. By virtue of its dependence from Claim 1, Claim 4 includes the features of Claim 1 distinguished from *Ng* above.

Claim 20 recites a computer-readable medium that carries instructions for performing the steps of the method of Claim 4, including the features distinguished from *Ng*.

As a result, Claims 4 and 20 are patentable over *Ng* under 35 U.S.C. § 102(e). The rejection of Claims 4 and 20 should be reversed.

E. The Features of Claims 5 and 21 Aren't Taught or Suggested by *Ng*

By virtue of its dependence from Claim 1, Claim 5 includes the features of Claim 1 that are distinguished from *Ng*. Therefore, Claim 5 is patentable over *Ng*.

Additionally, Claim 5 recites the feature, "**establishing a pointer from said instance to a property bundle**."

The Examiner cites element 310 of Fig. 3 and *iterator getOrdersForCustomer*() of Fig. 4B and associated text of *Ng* as allegedly disclosing this feature. The cited section of *Ng* refers to a hash table 310 where each entry in the hash table is a column object 312, 314 containing data for a particular field (*Ng*, col. 5, lines 64-66) and an iterator method to iterate through the orders for a particular customer. (*Ng*, col. 8, lines 44-46).

There appears to be nothing in the cited section that would correspond to establishing a pointer from an instance of a class to a property bundle. Indeed, the word "pointer" does not appear anywhere in *Ng*, which is significant because an iterator method is not a pointer, nor

does the use of an iterator method teach or suggest the use of a pointer between an instance of a class to a property bundle.

Therefore, *Ng* fails to disclose multiple features of Claim 5.

Claim 21 recites a computer-readable medium that carries instructions for performing the steps of the method of Claim 5, including the features distinguished from *Ng*.

As a result, Claims 5 and 21 are patentable over *Ng* under 35 U.S.C. § 102(e). The rejection of Claims 5 and 21 should be reversed.

F. The Features of Claims 6 and 22 Aren't Taught or Suggested by *Ng*

By virtue of its dependence from Claim 5, Claim 6 includes the features of Claim 5 that are distinguished from *Ng*. Therefore, Claim 6 is patentable over *Ng*.

Additionally, Claim 6 recites the feature, "**establishing a pointer from said attribute to said property bundle.**"

The Examiner cites elements *Class Order* 424, *int Order_id*, *Customer Customer_for_order* of Fig. 4B and associated text of *Ng* as allegedly disclosing this feature. The cited section of *Ng* refers to a class 424 and 2 attributes (an *int* and an object of Customer type). Specifically, "Class 424 includes data members order ID and date and also includes various methods to both set and get the values for these data members. Additionally, class 424 contains a field, Customer_for_Order, implementing the foreign key with a reference to the particular customer object that placed that order." (*Ng*, col. 6, lines 45-50).

There appears to be nothing in the cited section that would correspond to establishing a pointer from an attribute of a class to a property bundle. This is simply language describing a Class with 2 data members (variables), one being an integer and the other being an Object.

There is nothing in the cited section or the Examiner's assertions that could possibly qualify as teaching, disclosing or suggesting the establishment of a pointer to a property bundle.

Therefore, *Ng* fails to disclose multiple features of Claim 6.

Claim 22 recites a computer-readable medium that carries instructions for performing the steps of the method of Claim 6, including the features distinguished from *Ng*.

As a result, Claims 6 and 22 are patentable over *Ng* under 35 U.S.C. § 102(e). The rejection of Claims 6 and 22 should be reversed.

G. The Features of Claims 10 and 26 Aren't Taught or Suggested by *Ng*

By virtue of its dependence from Claim 1, Claim 10 includes the features of Claim 1 that are distinguished from *Ng*. Therefore, Claim 10 is patentable over *Ng*.

Additionally, Claim 10 recites the feature, "**maintaining a table that includes an entry that indicates that said instance is associated with said attribute**."

The Examiner cites the exact same elements of *Ng* as allegedly disclosing "maintaining a table that includes an entry that indicates that said instance is associated with said attribute" as recited in Claim 10 as the Examiner cited as to anticipate the "associating said attribute with said instance does not cause said attribute to become an attribute of said class" of Claim 1.

The only table maintained in the cited portions of *Ng* is an "order table," which is part of a database **schema** later translated into Java classes. Nothing in the customer table includes any entry that could possibly teach or indicate that an instance may be associated with an attribute not in the class, nor does *Ng* teach or disclose this feature. In fact, a review of Fig. 2 shows that the only fields in the customer table 202 cited by the Examiner are a customer ID column and a name column. While the Examiner apparently

analogizes the customer ID column and name column to attributes, no teaching is found to indicate how the order table is maintained to include an entry associating a particular instance with these alleged attributes. Fig. 4B shows a class 420 based upon the customer table, but there is no disclosure of how a table is maintained that associates a particular instance of the Customer class 420 with a particular value of an attribute where the attribute is not found in the Customer class.

Therefore, *Ng* fails to disclose multiple features of Claim 10.

Claim 26 recites a computer-readable medium that carries instructions for performing the steps of the method of Claim 10, including the features distinguished from *Ng*.

As a result, Claims 10 and 26 are patentable over *Ng* under 35 U.S.C. § 102(e). The rejection of Claims 10 and 26 should be reversed.

H. The Features of Claims 11-13 and 27-29 Aren't Taught or Suggested by *Ng*

By virtue of their dependence from Claim 10, Claims 11-13 include the features of Claim 10 that are distinguished from *Ng*.

Therefore, *Ng* fails to disclose multiple features of Claim 11-13.

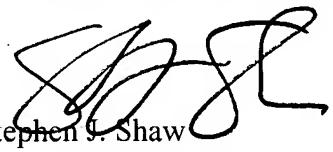
Claims 27-29 recite a computer-readable medium that carries instructions for performing the steps of the methods of Claims 11-13, including the features distinguished from *Ng*.

As a result, Claims 11-13 and 27-29 are patentable over *Ng* under 35 U.S.C. § 102(e). The rejection of Claims 11-13 and 27-29 should be reversed.

IX. CONCLUSION AND PRAYER FOR RELIEF

Based on the foregoing, it is respectfully submitted that the rejections of Claims 1-32 lack the requisite factual and legal bases. Appellants respectfully request that the Honorable Board reverse the rejections of Claims 1-32.

Respectfully submitted,
HICKMAN PALERMO TRUONG & BECKER LLP



Date: March 3, 2006

Stephen J. Shaw
Registration No. 56,442

2055 Gateway Place, Suite 550
San Jose, California 95110-1089
Tel: (408) 414-1224
Fax: (408) 414-1076

EXPRESS MAIL CERTIFICATE OF MAILING

Express Mail" mailing label number EV 861821695 US Date of Deposit: March 3, 2006

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to Mail Stop Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

On March 3, 2006 By _____
Jennifer Newell

CLAIMS APPENDIX

1. A method of specifying a structure, within a computer system, of an instance of a class, the method including the step of:

associating with said instance of said class an attribute that is not in said class or any superclass of said class, thereby establishing for said instance said structure that includes storage for data associated with said attribute; and

wherein associating said attribute with said instance does not cause said attribute to become an attribute of said class.

2. The method of claim 1 wherein the step of associating further includes the steps of:

establishing a property bundle that is associated with one or more attributes that are not in said class or any super class of said class; and

associating said instance with said property bundle.

3. The method of claim 2 further comprising the step of:

storing within a database, objects that define said instance, said property bundle, and said one or more attributes.

4. The method of claim 1 further comprising the step of:

maintaining an object relational mapping system that indicates a correlation between said instance and data stored in a relational database.

5. The method of claim 1 wherein said step of associating includes establishing a pointer from said instance to a property bundle.

6. The method of claim 5 wherein said step of associating includes establishing a pointer from said attribute to said property bundle.

7. The method of claim 1 wherein a property class contains said attribute.

8. The method of claim 1 further including the step of:
associating a key with said attribute wherein said key identifies said attribute.

9. The method of claim 8 wherein said key is a user defined key.

10. The method of claim 1 wherein the step of associating further includes the step of:
maintaining a table that includes an entry that indicates that said instance is associated with said attribute.

11. The method of claim 10 wherein the step of maintaining a table further includes the step of:
maintaining said entry to include a key that identifies said attribute.

12. The method of claim 10 wherein the step of maintaining a table further includes the step of:
maintaining said table externally to said instance.

13. The method of claim 10 wherein the step of maintaining a table further includes the step of:
maintaining said table internally to said instance.

14. The method of claim 1 wherein the step of associating further includes the steps of:
storing into said instance a hash table; and

locating an entry in said hash table for said attribute.

15. The method of claim 14 further includes the steps of:
receiving data that is designated as a key for locating said entry in said hash table; and
using said data as said key to locate said entry without determining whether said data conforms to software rules.

16. The method of claim 1 wherein the class is a file type and said instance is a file of said file type in a file system wherein the step of associating includes associating with said file of said file type an attribute that is not associated with said file type or any super class of said file type.

17. A computer-readable medium carrying instructions for specifying a structure, within a computer system, of an instance of a class, the instructions including instructions for performing the step of:
associating with said instance of said class an attribute that is not in said class or any superclass of said class, thereby establishing for said instance said structure that includes storage for data associated with said attribute; and
wherein associating said attribute with said instance does not cause said attribute to become an attribute of said class.

18. The computer-readable medium of Claim 17 wherein the step of
associating further includes the steps of:
establishing a property bundle that is associated with one or more attributes that are not in said class or any super class of said class; and
associating said instance with said property bundle.

19. The computer-readable medium of Claim 18 further comprising instructions for performing the step of:
storing within a database, objects that define said instance, said property bundle, and said one or more attributes.

20. The computer-readable medium of Claim 17 further comprising instructions for performing the step of:
maintaining an object relational mapping system that indicates a correlation between said instance and data stored in a relational database.

21. The computer-readable medium of Claim 17 wherein said step of associating includes establishing a pointer from said instance to a property bundle.

22. The computer-readable medium of Claim 21 wherein said step of associating includes establishing a pointer from said attribute to said property bundle.

23. The computer-readable medium of Claim 17 wherein a property class contains said attribute.

24. The computer-readable medium of Claim 17 further including instructions for performing the step of:
associating a key with said attribute wherein said key identifies said attribute.

25. The computer-readable medium of Claim 24 wherein said key is a user defined key.

26. The computer-readable medium of Claim 17 wherein the step of associating further includes the step of:

maintaining a table that includes an entry that indicates that said instance is associated with said attribute.

27. The computer-readable medium of Claim 26 wherein the step of maintaining a table further includes the step of:
maintaining said entry to include a key that identifies said attribute.

28. The computer-readable medium of Claim 26 wherein the step of maintaining a table further includes the step of:
maintaining said table externally to said instance.

29. The computer-readable medium of Claim 26 wherein the step of maintaining a table further includes the step of:
maintaining said table internally to said instance.

30. The computer-readable medium of Claim 17 wherein the step of associating further includes the steps of:
storing into said instance a hash table; and
locating an entry in said hash table for said attribute.

31. The computer-readable medium of Claim 30 further including instructions for performing the steps of:
receiving data that is designated as a key for locating said entry in said hash table; and
using said data as said key to locate said entry without determining whether said data conforms to software rules.

32. The computer-readable medium of claim 17 wherein the class is a file type and said instance is a file of said file type in a file system wherein the step of associating includes associating with said file of said file type an attribute that is not associated with said file type or any super class of said file type.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.